PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 :
G06F 11/00

A2

(11) International Publication Number: **WO 98/45778**

(43) International Publication Date: 15 October 1998 (15.10.98)

(21) International Application Number: PCT/IL98/00170

(22) International Filing Date: 8 April 1998 (08.04.98)

(30) Priority Data:
120632 8 April 1997 (08.04.97) IL

(71)(72) Applicant and Inventor: ZUTA, Marc [IL/IL]; Ben Yehuda Street 19, 49373 Petah Tikva (IL).

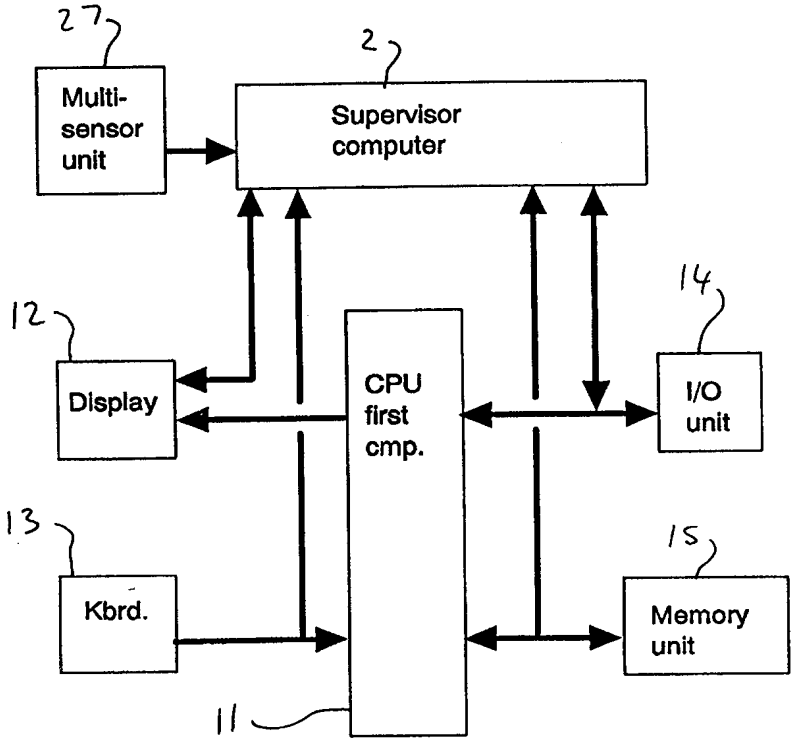
(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, GW, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: ANTIVIRUS SYSTEM AND METHOD

(57) Abstract

In an antivirus multiprocessor system, a second processor attached to a first processor for monitoring its performance and intervening if predefined behaviour thereof is detected. The second processor includes means for continuously supervising the operation of the first processor to detect virus–related activities therein by comparing actual performed instructions in the first processor with instruction sequences corresponding to known viruses or to predefined suspect behaviour. It will stop the first processor after a virus detection. A plurality of sensors is used to detect suspect activity in various media like radio frequency or wireless RF, serial or parallel communication channels, ultrasound, sonic or subsonic waves, power lines and/or optical media including infrared, visible light and/or ultraviolet. An antivirus protection method includes the steps of: A. connecting a second processor to an applications processor; B. using the second processor to continuously monitor the activities on the first processor and to compare to predefined forbidden activities; and C. taking actions in real time when a forbidden activity takes place, to prevent damage to the applications processor.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

Antivirus System and Method

Technical Field

This invention concerns antivirus protection systems and methods. The invention relates in particular to such systems in which a second (supervisor) processor is added to a first (applications) processor to continuously monitor its operation so as to prevent damage from viruses or software errors.

Bakground Art

With the proliferation of personal computers and of software therefor, there is a tendency for the hardware to become more and more powerful, and for the software packages to become more and more complex. Software packages nowadays take large disk space, in the range of megabytes to tens of megabytes each.

Heretofore, in this new environment, the computer has become less "personal", in the sense that the user cannot supervise the operation of the computer, or have a reasonable measure of assurance about what the computer is actually doing.

The result is that there are occurences of the computer not doing what its owner intended it to do. Rather, sometimes, the computer is doing something else, which causes damage to the user. The user usually learns too late of the damage, and sometimes may not know about it at all.

The very advantages of the computer have become its greatest weakness: Since the computer is fast and powerful, much damage can be done in a very short time. Before the user can intervene, in a time span of miliseconds, a virus can delete hundreds of files and megabytes of memory.

Presently used software packages are very large, including hundreds of files each, in tens of subdirectories, and taking tens to hundreds of megabytes. It is practically impossible for the user to verify all these files to detect a virus or to assess the damage done to these files.

Computers are used for communications. Much information and many programs are available in the INTERNET, for example. How is one to be sure that the information or programs are not contaminated?
The widely used E-Mail messages may include viruses therein.

The problem is much aggravated in computer networks. A network includes tens to hundreds of computers, each vulnerable to attack and capable of transmitting a virus to other computers. Moreover, a network is usually open to access from the outside, so that a malicious person can connect to a network without physically breaking into the victim's facilities. Much harm can be done from far away.

Following are several examples of computer malfunction instances.

1. Computer viruses. These are programs maliciously introduced into the user's computer, with the express purpose of causing damage.

Sometimes the virus erases information stored in the computer. Other times the virus alters the information without the user's permission. Viruses are known to create dummy files or enlarge existing files, which clutter the memory storage unit, so that the user may not have enough memory left for their purposes.

The name "virus" apparently came from a property of these programs to copy themselves to other media like diskettes, and therefrom to other computers, so that the "virus" is actually "spreading" to "contaminate" other computers.

One approach to the computer virus is the "antivirus" software, which stores digital patterns, each characteristic to a known virus. When starting the computer or inserting a diskette, the antivirus program scans the memory and compares the patterns therein with the stored patterns.

An alarm is activated if the patterns correspond, this indicating the presence of a virus.

The big problem with this approach is that only known viruses, that is viruses whose characteristics are stored in the program, can be detected. It appears that new viruses are continuously created, which are not detected nor protected against by the antivirus programs. In this war between virus and antivirus, the virus will always win, by its time lead. Thus, antivirus programs cannot prevent damage from new viruses which are not included therein.

Another problem with this approach is that the memory is only scanned prior to its use. A self-changing virus can become active after it was scanned, while the antivirus is no longer active.

If an antivirus program is used concurrently with the application, then there is a deterioration in performance, since the same processor is used to alternately perform instructions pertaining to the antivirus and the application. Thus, the application will take more time to run.

Moreover, antivirus programs do not indicate the source of the virus, that is which program contains it. At present, the user does not know which program caused the damage or inserted the virus, thus action cannot be taken at source, to stop that source of viruses and/or to claim damage compensation. The required evidence to prosecute the culprit is not available.

Antivirus programs do not help the user regain control of his/her computer. The program only demands to "clean" the virus, without reporting to its user which files are contaminated and what are the consequences of the cleanup, or allowing any choices to the user.

It is believed by the present inventor that software means cannot be used to protect from another software. A virus can tamper with the antivirus program itself, so that further actions by the user and/or updates of the antivirus are meaningless and useless.

Brute force approaches like denying access to disk or forbidding write operations make the computer practically useless, since a large part of the activity with computers is to read/write files and operate on information. Such a measure may be taken for a limited time. When the user can take it no more and disables the protection, the computer is completely vulnerable to attack.

2. Backdoors, for example a communication program including a routine to allow unauthorized access to another's computer by an outsider. Remote diagnostic programs are a needed feature in computers and networks; the same programs can be used by unauthorized people as backdoors to gain access to the victim computer.

Thus, unauthorized person A can remotely connect to the computer of user B. The trapdoor in user's B computer, of which B is not aware, allows A to access confidential files in that computer, to read them or make changes or to otherwise do damage. It is even possible for user B to connect to a remote computer, without that user being aware of the fact that the other party may use that trapdoor during that same session.

Trapdoors can also be used to introduce viruses, possibly with delayed action, or to access privileged information.

It was stated that most of the damage to computers to any firm is done by employees or former employees at that firm.

The inside knowledge gained during employment there may be used to bypass safequards and do damage.

3. Undesired TSRs (Terminate and Stay Resident programs).
A personal computer, during the several seconds it takes to prepare itself at turn-on until it is ready to accept user's commands, may perform tens to hundreds of millions of instructions. Some programs instruct the computer to be left and active in the memory, thus influencing the computer operation thereafter. This is basically a desirable action, for example a TSR to detect the activation of the mouse and to respond accordingly.

Unfortunately, with the advent of viruses, some TSRs come to perform undesired functions. For example, a maliciously introduced TSR may slow down the computer, by repeatedly performing an unnecessary piece of useless code, without user's knowledge.

The end result is that the computer performs slower than is to be expected, thus preventing the user from getting the full performance from the computer.

4. Unorganized or conflicting TSRs .
Nowadays a personal computer contains tens of different programs, each with its own TSR, all competing for the same computer resources and giving sometimes contradicting instructions to the computer. No software provider takes responsibility for the coexistence of the various programs in the computer.

The user is helpless and cannot intervene, since he/she does not know what is actually going on inside the computer.

5. Uninstall of software packages. It is sometimes difficult to remove a package which is no longer desired. Today software packages are large, tens of megabytes each, with hundreds of files being created on disk and changes performed on existing files. It is sometimes difficult to remove files scattered all over a large disk and to undo changes to existing files.

The problem is that the user sometimes is not completely aware of actions performed by the software package during installation. Sometimes, even with the most friendly install routine, the user will not bother to record all the actions performed, for each package installed, this being a tedious task. Human factors are as important as technical ones, and should be given careful attention.

Moreover, means for protection agains viruses should provide reliable protection, without leaving any doubt as to whether the protection device itself is secure or was compromised by a virus.

The protection means should provide the desired protection without a sizable degradation in the overall performance of the computer.

The protection means should perform its tasks without demanding too much of user's attention or intervention. Eventually, any protection means which is not easy to use will be disposed of.

It is an objective of the present invention to provide for an antivirus system and method using a first and a second processor means for overcoming the abovedetailed disadvantages.

Disclosure of Invention

It is an object of the present invention to provide an antivirus system and method using a multiprocessor system comprising a first processor for performing application programs and a second processor for continuously supervising the operation of the first, to protect it from misuse.

The object is achieved by an antivirus system as disclosed in claim 1.
In accordance with the invention a second processor, the supervisor, is attached to the first processor, the application processor, so that the supervisor includes means to continuously monitor the activity in the application processor, to detect viruses and abnormal operations.

The supervisor further includes means for intervening to stop the applications processor and/or to issue a warning when the virus is detected, to prevent damage.

According to a second object of the present invention, the supervisor itself is protected from attack. It has its own computer and memory, which are independent from the application processor and are protected from any attack by a virus in the application processor. Additional means protect from physical attack.

According to a third object of the present invention, the supervisor computer includes a combination of fast monitoring means in hardware, with smart verification means in a programmed device. These means operate together to protect from known viruses as well as from unknown viruses according to their suspect behavior.

According to a fourth object of the present invention, the supervisor method of operation allows both for detection of viruses in real time, and for smart monitoring of sensitive operations without a sizable decrease in performance. Operation at several levels of complexity is disclosed.

According to a fifth object of the present invention, the supervisor performs a smart verification method, to detect potentially dangerous activities.

According to a sixth object of the present invention, means are detailed for the supervisor to detect viruses without slowing down the applications processor.

According to a seventh object of the present invention, the supervisor includes a multisensor unit which monitors everything going on, in and around the application processor, including a plurality of sensors for sensing the various effects indicative of the presence of a virus.

Further objects, advantages and other features of the present invention will become obvious to those skilled in the art upon reading the disclosure set forth hereinafter.

Brief Description of Drawings

The invention will now be described by way of example and with reference to the accompanying drawings in which:

Fig. 1 is a simplified functional diagram, illustrating the attachment of a supervisor computer to an application computer.

Fig. 2 is a simplified electrical diagram illustrating the attachment of a supervisor computer to an application computer.

1

Fig. 3 details the functional structure of supervisor computer

Fig. 4 details the structure of real time monitoring means

Fig. 5 details the omparator structure and operation

Fig. 6 illustrates the structure of monitoring means using digital signature

Modes for Carrying out the Invention

A preferred embodiment of the present invention will now be described by way of example and with reference to the accompanying drawings.

Fig. 1 illustrates the attachment of supervisor computer 2 to an application computer. The application computer, in this simplified functional diagram, includes a Central Processor Unit CPU 11, connected to display means 12, keyboard 13, Input/Output unit (I/O) 14 and memory unit 15.

The application computer is the original computer which provides the computer functions desired by the user, and to which is added the supervisor 2 to ensure that indeed it will perform the desired functions, without the interference of viruses or malevolent programs.

One function of the supervisor 2 is to monitor activity on busses of CPU 11, like data, control and address between CPU 11 and memory 15, and input/output activity between CPU 11 and I/O means 14. Supervisor 2 connects to these busses to watch instructions and check if some forbidden activity is about to take place. It searches for known viruses patterns or privileged instructions which require authorization.

The monitoring is done concurrently with normal operation of the application computer, without slowind it down. Supervisor 2 reads commands and data "on-the-fly" in real time.

Another function of supervisor 2 is to act to stop or prevent forbidden activity, after detecting a virus as mentioned above. This is done by: activating a signal to CPU 11 to Reset processor or bring it to Wait state or stop CPU 11 and put its activities under closer scrutiny or log the activities performed.

A third function of supervisor 2 is to perform protected, direct dialog with the user, using the application computer resources like the keyboard 13 and display 12. Supervisor 2 has direct access to these devices, and it can also immobilize the CPU 11 during the dialog with the user. This is used to reliably notify the user of virus problems, to ask questions regarding permitted operations and to accept instructions from the user, like the subdirectories it is permitted to read and/or write files therein.

Moreover, the information in computer 2 can be updated by reading from a diskette (not shown), for example to update the list of viruses. Again, this is preferably done while CPU 11 is not active, to avoid interference with, or eavesdropping to, these activities by a potential virus or ilegitimate program in the applications processor.

The applications computer is deactivated using, for example, its RESET or HOLD or WAIT while supervisor 2 is using the resources therein like the diskette or keyboard or I/O channel.

Thus, the functions of supervisor 2 is to watch and report, prevent disaster, help recover from problem, let the user know how disaster happened, and who caused it.

The multisensor unit 27 supports the monitoring function of supervisor 2, by providing additional information relating to possible effects of viruses.

Thus, a virus can establish a communication link with an unauthorized outsider using any of a wide variety of media, for example through a wireless or RF link, ultrasonic or sonic or subsonic waves, signals on the power lines etc. High frequency signals can be transmitted over power lines to outside the firm or the user's home.

Other possible links may include existing serial or parallel communication channels, like an RS-232 link to a modem or a local net or a phone dialer. These links may be activated under a virus control without the permission or knowledge of the legitimate user. Other possible links may include optical media including infrared, visible light and/or ultraviolet.

Unit 27 includes a plurality of sensors for monitoring the various media for any potential undesired activity. The user may not be able to otherwise sense these activities which may go on about their computer.

Thus, the antivirus system and method uses a first processor, including CPU 11 and related units, for performing application programs and a second processor 2 for continuously supervising the operation of the first, to protect it from misuse. The supervisor 2 is attached to the application processor, wherein supervisor 2 includes means to continuously monitor the activity in the application processor, to detect viruses and abnormal operations. The monitoring is done in real time.

The supervisor further includes means for intervening to stop the applications processor and/or to issue a warning to user when the virus or abnormal operation is detected, to prevent damage to the application processor.

Fig. 2 details the electrical attachment of a supervisor computer 2 to an application computer, in a simplified block diagram.

The supervisor computer 2 connects, or couples to a first processor, that is the applications computer. The multisensor unit 27 gives the supervisor 2 the ability to monitor the various media around the applications computer, as detailed below.

The applications computer includes the CPU 11, display means 12, keyboard 13, Input/Output unit (I/O) 14, memory unit 15 and other resources 16. All these devices are connected to each other through control/data/address bus 17, as known in the art.

Supervisor 2 connects to bus 17, thus gaining access to all the resources of the application computer. Whereas supervisor 2 monitors activity on the applications computer through bus 17 and exercises control over the application computer if need be, it does not use the memory 15 of the applications computer to store its programs or data.

By including (not shown) its own, separate processor and memory with programs and data, supervisor 2 is protected from attack by virus in the applications computer.

Supervisor 2 includes means to protect from physical attack, tampering with, removal from bus 17. Its board (not shown) is encased in hard plastic, to prevent access to its components. Mechanical means may be included (not shown) to lock to motherboard or bus 17 of applications computer.

Additional physical protection means include the supervisor 2 board being soldered to the connector of bus 17, or the supervisor being produced as an integral part of the motherboard of the applications computer.

In another embodiment, the supervisor 2 is manufactured as an integral part of the CPU 11. This achieves a CPU which is protected from viruses, and is practically impossible to separate between the actual CPU and its protection means.

Thus, the supervisor 2 itself is protected from attack. It has its own (not shown) computer and memory, which are independent from the application processor and are protected from any attack by a virus in the application processor. Additional means protect from physical attack.

Fig. 3 Illustrates one embodiment of the functional structure of the supervisor 2 mentioned in Figs. 1 and 2 above. The supervisor includes a combination of fast, real time monitoring means including units 22, 23, 24 and a slower, smart unit comprising controller 21.

Following is a detailed description of the structure and operation of the supervisor.

The supervisor structure includes the following means:

1. monitoring means, including registers 22 and 23 and comparators 24, to evaluate operation of first processor in real time for detecting abnormal operation, to stop first computer and issue alarm or warning or indication to controller 21. A shift register 22 continuously receives and shifts in real time instructions read in the first, or the supervised, computer. A plurality of comparators 24 each compares the instruction string with the string of a known virus or a predefined suspect behavior, to detect viruses in real time.

2. controller means 21 to: initiate the monitoring means at start-up, take actions when alarm is issued, dialog with user, program/update internal memory, log and analyze activities performed prior to an alarm or a virus attack. A smart analysis is done to detect suspect behaviour, indicative of virus, after receiving indication from monitoring means.

3. internal memory 25 – for virus patterns and forbidden/ privileged instructions, log of activity prior to virus, allowances in each application program (different set of allowances and masks for each application, according to user's instructions).

The memory can be implemented in the controller IC itself or in a separate IC or ICs or on disk, if adequate safeguards are taken to protect that disk area.

4. multisensor unit 27

The bus 17 of the applications computer, including control, data, address is coupled to comparator shift register 22. Various embodiments are possible, with corresponding levels of performance and circuit complexity. The simplest is to couple only the data bus to shift register 22, using part of the control signals to strobe the data in at the correct timing, as known in the art.

Control signals on bus 17 indicate when an instruction is on the bus, whether this is a memory or I/O operation, whether it is a Read or Write. Register 22 can be wired accordingly to accept a new sample for each new instruction, or only for memory read, or for all data bytes on the bus. This embodiment is suitable for recognizing instructions and various routines.

A more complex embodiment reads into register 22 the data together with the address and the control signals. This enables to search for instructions relating to specific addresses in RAM, like reading from the interrupt vectors area or the DOS sections, or writing to these locations, which are privileged locations. This embodiment requires a wider register 22, that is each stage contains more bits.

Register 22 includes a plurality of stages (not shown), each k bits in parallel. As a new sample is read in from bus 17, all the previous samples are shifted one stage to the right. The oldest sample is transferred out of register 22, to log shift register/FIFO 23.

The samples in the various stages of register 22 are output to extended processor bus 220. Thus, bus 220 contains the data bus and optionally also control and address, for a plurality of times: the last sample, the sample a clock ago, then two clocks ago etc...to 100 or several hundred samples.

Bus 220 contains the words or bytes for the last instructions on the bus 17. These instructions are compared in comparator unit 24 with the instructions for a specific virus, or the instructions sequence for a sensitive operation which demands scrutiny by contoller 21. The monitoring unit contains a plurality of comparator units 24, each programmed to detect one virus or privileged instruction, as detailed below. All the comparators 24 operate concurrently in real time. Each comparator 24 includes a string of a specific virus or instruction sequence, and possibly an ignore mask to ignore irrelevant parameters in the instruction string.

The comparators 24 issue an alarm/warning/action on bus 28, which includes busses 241 and 247. Thus, alarm/warning /action bus 28 is used to issue the alarm/warning if a suspect string or instruction was detected. The bus 28 combines with similar busses from the other comparator units, all connected in parallel.

Bus 28 contains action signals, comprising command bus 247, to take immediate action (fast) if a dangerous string is detected, to RESET or HOLD in wait the applications computer for example. Bus 28 also includes the report alarm bus 241, to report to controller 21 that an alarm/warning or other virus-related activity took place. The specific word on bus 241 can indicate the type of action or virus which was detected.

Thus, according to preprogrammed parameters, when an expected instructions sequence is detected in one of the comparators 24, then the application computer can be immediately stopped if necessary, and the controller 21 notified. If a less dangerous instruction is detected, then controller 21 can be notified without stopping or resetting the applications computer.

Supervisor controller 21 includes nonvolatile memory means 25 holding its programs and parameters, like the virus list. This information is protected from access from the applications computer through bus 17. Thus, programs in supervisor computer are protected from access from outside. The programs for controller are not changeable, to prevent tampering with. No software or virus in PC can affect the second processor. No event in PC can change the programs in second computer, including the user or a virus or any program or occurence in PC.

The internal storage is tamper-proof by application processor. Preferably the programs, parameters (whatever influences operation) are stored in internal ROM/PROM/ CMOS RAM/ Flash Memory.

Time/ date unit 26 serves to perform time-sensitive functions. For example, controller 21 can check the number of different files addressed by an application per second. Too many files acessed in a short time is a suspect activity, suggesting a virus. Time can also be used to log activities, to provide evidence for subsequent legal action.

Controller 21 also reads the report activity bus 231, to read past instructions from register 23, so as to reconstruct activity prior to alarm.
Log shift register/FIFO 23 can be longer than register 22, since only the final stage output goes to an output bus 231. It can keep log of thousands to hundreds of thousands of the last instructions. This allows controller 21 to inspect and find what events took place prior to the virus attack, or where the virus come from.

Controller 21 can take direct control over the application computer through bus 217. This bus can include (not shown) lines to Hold that computer inactive, or to Reset it, together with control, data and address lines to gain access to the application computer and its resources, for example using DMA.

Load parameters bus 29 is used by controller 21 to load parameters into comparators 24, each with a different instruction or instruction string and a corresponding "ignore" mask for each byte or bit, and a desired action to take if that string is detected.

It is possible to update the information/parameters in memory 25, while updates are done under controller 21 supervision. Encrypted messages may be used to program new virus patterns. It is preferred not to allow programming controller while the application is operational, but only with applications stopped by controller 21.

In another embodiment, monitoring means includes means (not shown) for computing digital signature or CRC (Cyclic Redundancy Code) and comparing with signature of viruses, instead of directly comparing bytes. This saves hardware. The embodiment is illustrated in Fig. 6. It is possible (not shown) to use one comparator, while multiplexing virus signatures and masks, to save hardware.

Another embodiment (not shown) uses a DSP (Digital Signal Processor) means or other programmed device instead of hardware, to compute signatures or directly compare with string of bytes and/or to perform masking operations. Very fast processors may be used to perform comparison with virus string as detailed above using a microprogrammed implementation of the abovedetailed structure and its operation method, as detailed with reference to shift register 22 and the comparators 24.

Thus, the supervisor computer includes a combination of fast monitoring means in hardware, with smart verification means in a programmed device. These means operate together to protect from known viruses as well as from unknown viruses according to their suspect behavior.

The controller 21 performs the following functions, in addition to the smart verification which is detailed in the following chapter:

1. Loader, at power-up loads all patterns and ignore to all registers from a mass storage like a Flash Memory.

2. Programmer to accept updates, check validity, then write update in nonvolatile memory. Memory is nonvolatile, like flash or cmos or similar, or magnetic. Internal in supervisor.

3. Update programs, data in internal memory. Can be changed only when entered with special format and CRC and/or encrypted, to enable update of virus list, programs etc by responsible provider. Responsible to user as to integrity of program. PC is in reset or wait during this update, so no virus or nonhonest user can watch, record or tamper with program update in second computer.

Supervisor Computer Operation

There are several possible embodiments, from the simple to complex. These are made possible with the structure detailed above, with monitoring unit including registers 22 and 23 and comparators 24 (fast comparison) under the supervision of the controller 21 (smart analysis):

1. Simple operation, virus pattern detection. At power-up the controller 21 loads known viruses into the monitoring means hardware, that is in comparators 24.

During the subsequent operation of the first computer (the application computer or PC or computer network), the monitoring means act to continuously detect viruses and stop the PC to prevent damage. Controller 21 is inactive, except to display the log of instructions prior to alarm, to report to user what happened there.

2. Guided scan protection: the monitoring means is programmed to detect the activity in the application PC of a piece of code being loaded to RAM for execution. When this activity is detected, the application CPU is stopped and the controller 21 is prompted accordingly. The controller 21 then reads all the instructions which are in RAM and ready for execution in the PC, just prior to their execution, by using bus 217.

While the instructions are thus read, the monitoring means reads samples through bus 17 into register 22 as detailed above and searches in real time, in hardware, for all the known viruses which are programmed therein. Thus, even viruses with self-changing code are detected, since this method verifies each piece of code just prior to its execution.

3. Two-stage protection: comprises fast real-time verification using the monitoring monitoring means, and a higher level, slower, smart verification using the controller 21, for sensitive operations.

At power–up the controller 21 loads known viruses pattern as well as sensitive operations which demand further scrutiny, like interrupts or file operations or I/O. The smart verification is detailed below.

During normal operation thereafter, if a virus is detected then the monitoring means act promptly to stop the applications CPU through bus 247, to prevent damage. If a sensitive operation is detected in the monitoring unit, then the CPU is temporarily stopped and the controller 21 is prompted through bus 241 to perform a smart, in–depth analysis of the situation.

This may be done in addition to controller 21 acting to report virus activity, as for the simple operation detailed above. This embodiment includes the simple hardware protection and further, the second level of verification/protection.

Advantage: protects from attack by yet unknown viruses, with unknown patterns, by detecting suspect activities. These are activities expected from viruses, like writing to files, changing files and more.

Rational: smart tests are difficult to perform in hardware, but are easier to implement in software, in the programmed controller. This, however, takes time and cannot be performed in real time, while the PC operates as normal. According to the present invention, the smart tests are performed for sensitive operations in the PC, which operations are slower than the rest. For example, disk operations are slow since they involve the mechanical movements of the disk head and platter, and take time in the order of milliseconds.

The PC is preferably stopped during the smart test performance.
Thus, the additional time it takes to perform the smart test has no considerable effect on the overall performance of the PC, since smart test can be peformed in about 10 – 100 microseconds, this time being so much shorter than the sensitive operation in the PC, that the overall performance is not much affected.

The smart test can be peformed is such a short time, for example using a 16-bit controller like the Intel 80196.
Similarly, I/O operations like a modem have a relatively slow bit rate, and take overall time in the order of seconds to minutes. Thus, a smart verification prior to the actual performance of the I/O function has no sizable effect on overall PC performance.

4. Active protection: may be performed in addition to the two-stage method. The controller 21 initiates activities in the PC, which are expected to stimulate or provoke viruses into action. Thus the virus is detected, by its activity.
The controller creates a controlled environment, wherein the activity of the PC can be completely predicted; any deviations from the expected activity are the result of a virus, thus the virus is detected before it has a chance to attack.
This is an offensive method of operation, wherein the protection means strikes first to provoke the virus, then to detect it and neutralize it.

Thus, the supervisor method of operation allows both for detection of viruses in real time, and for smart monitoring of sensitive operations without a sizable decrease in performance. Operation at several levels of complexity was disclosed.

Smart Verification Method

This method is performed by controller 21, after it is prompted through bus 241 that a sensitive instruction or sequence of instructions took place. It includes various routines for analyzing the activities in the PC from several aspects.

1. Monitors execution since init or reset. Logs TSR loading, disk access, communications access, especially in the initial stages of the application computer or PC operation. Logs autoexec.bat and config.sys execution, with the actual performance taking place.

2. Pays attention to special instruction or access to specific locations like access to directory or to boot sector, writing to exe, com , bat or sys files, reading too many different files in a short time, accessing files outside a permitted subdirectory, writing to files without some processing or user input (without apparent reason or prompting to do so).

3. Logs install or setup performance, to allow delete install

4. Logs TSR installation, and verifies TSR execution in actual life

5. Tracks virus to source; reports which program initiated, introduced virus in our PC, to take legal action again those responsible for virus. Logging the virus activity and tracing it to source can provide the required evidence to prosecute the culprit.

6. Communicates with user through keyboard, display, diskette, I/O

7. Keeps records in PC disk. May encrypt to preserve on disk. Prevents PC from access to those areas, like from a virus.

8. Logs creation of files on disk. In previous art computers, various applications create *.tmp, or temporary files, of size 5 – 15 megabytes, which clutter disk. If deleted, may interfere with applications. If this is malicious – user may want to know which program did it. If it is an error by programmer – user may want to know , to ask why there are retainted files to clutter disk. A notification may be issued like:

"you forgot a file or 100 files ... in my computer.." .

9. Reports to user by RS–232 or other I/O, while keeping PC inactive to prevent a virus therein from tampering with the transfer or to log it, for later misuse of information, keys or communication protocol.

10. Define illegal activities, or those which require user's manual approval. If detected– stop it and report. for example– request of information by LAN. Prompt from LAN or I/O or Internet to read file or to change it or delete information.

11. Read part of each TSR loading at init/reset into comparator units, to check how often they execute. If it is done too ofter– maybe virus. Report all TSRs being loaded; by whom; memory RAM occupied.

12. Log, monitor disk/diskette access. Too often is unnecessary, results in wear and tear of mechanical parts. Maybe reading information which is not really necessary, or doing damage to many files.

13. Log, monitor access to too many files at once, during short time period. In normal use, computer works with limited number of files. Monitor access to too many subdirectories. User can limit operation to several or few subdirs or to specific areas of disk.

14. Define forbidden combination of activities: ie communications + access to specific subdirs. or comm + printing. or comm + access to other than one subdir, where we deposited a file to read or write. Then load these to comparator units 24, to report back to controller 21 if detected.

15. Special adaptive comparator unit (or several such): freezes a pattern of executed instructions from SR 22 to registers, to compare with itself later on. Detects a virus which creates dummy loops, repetitive execution of unnecessary programs. Method of use: From time to time, at random, controller 21 reads pattern of instructions now executing through bus 231, and load through bus 29 into comparators 24.

The comparator then checks for repetitions of this patterns, and reports to controller. If the pattern repeats too often, it may be a virus. If not, then after a set time a new instructions string is read at random and compared with thereafter.

This solves the problem of a virus slowing down computer operation by initiating it to do unnecessary, dummy commands. It performs an adaptive monitoring and detection of unnecessary repetitions of code.

16. Correlator between program executing and disturbances monitored by sensors (multi-sensor unit 27). If there is correlation – stop and report to user. The user may be unaware that activity in his/her computer are a result of commands received from the LAN or Internet or a bug operated by radio for example, or with ultrasonic waves.

Thus, the supervisor performs a smart verification method, to detect potentially dangerous activities.

Fig. 4 details the structure of real time monitoring means in one embodiment, in simplified block diagram. This is the monitoring means hardware, used to monitor without slowing down the PC or application computer.

Data multiplexer and latches means 222 is used to read information from the applications computer bus, including data bus 172, control bus 173 and address bus 174. These are part of control/data/address of applications computer 17. Means 222 may read all or part of these busses. The PC bus uses data, address, control on 8 or 16 or 32 bit bus. Means 222 aligns the bus by bytes for proper analysis of programs execution, or brings into other uniform form the input samples.

The result is samples of instruction/data bus 223, which are transferred to comparator shift register 22. After input, these samples are shifted to present an instructions sequence for the comparators 24, to compare with existing viruses.

Bus 223 may contain, in addition to data and commands, also the status, strobe etc– the CPU controls issued on the bus. This provides the context of data issued or read: rd/wr, IO/memory, bus width.

These are specific to each CPU like the Intel 386 or Pentium, Motorola 68020, Analog Devices DSP 21020, 21060, and should be interpreted as such.

The shift register includes two parts:
comparator shift register 22, including stages like 224, 225. It shifts the data, commands all the time to the right, then to output bus 229;
and
log shift register/FIFO 23 , which shifts all the time as well, with its output bus 231 connected to controller (not shown).

Register 22 is used for comparison with patterns to search. It is small, about 100– 200 samples with outputs at every stage/sample (internal to supervisor). The first output is the data bus latest output C0 226, with subsequent outputs for previous samples. Thus, data bus i–th output Ci 227 represents one of the n stages in SR 22, and data bus earliest output Cn 228 is the oldest sample in register 22. The busses 226, 227, 228 comprise the abovementioned bus 220.

Register 23 keeps log of last operations, to track virus to source, by reconstruction of activity performed prior to virus attack/detection large, about 10 – 100 ksamples. Contains only output total 231, to controller 21.

Output busses from register 22, including bus 226, 227, 228 are connected in parallel (the same data) to a multitude of comparator units 24, each including:

1. instruction/data reference register 31, which is fixed after initial loading from controller. It includes a plurality of stages, corresponding to the stages of register 22, and with output bus at each stage, of which are shown the reference bus latest value R0 316, the reference bus i–th value Ri 317 which represents one of the n stages in register 31, and the reference bus earliest value Rn 318. The load reference bus 291, part of parameters bus 29, is used to load the initial values into register 31 at power–up.

This programmed structure allows for flexible operation, to load various viruses or instruction strings to be searched for.

2. a plurality of comparator stages in parallel. Illustrated is the comparator first stage 331, for last instruction value. It compares latest output C0 226 with corresponding reference R0 316, using AND mask from bus M0 326. The comparison is bitwise, that is each of k bits of bus C0 226 is compared with a corresponding bit of bus R0 316, with mask bit from M0 326 being applied to allow or ignore the result, then all the partial bit results are collected to result in result of comparison 361, which is detailed below.

Similarly the comparator stage 333, is used for earlies instruction value. Compares earliest output Cn 228 with corresponding reference Rn 318, using AND mask from bus Mn 328. The result of comparison 363 is the output.

AND gate 334 sums up all the results of comparison 361, 363 and all the results in between (n in all), to issue alarm strobe 335 if all the comparison results are positive, indicating the complete detection of a virus string.

3. Mask register 32 . Is fixed after initial loading through load mask bus 292, part of parameters bus 29, at power up. Like register 31, there are outputs at each stage, illustrated with mask bus latest value M0 326, mask bus i-th value Mi 327 represents one of the n stages in register 32, and mask bus earliest value Rn 328.

4. action register 34 holds a digital value indicating the type of action to take, which is related to the virus or event stored in register 31. The action may include Reset the CPU, Hold or only a warning to the controller 21 (see Fig. 3) of the supervisor processor. It is loaded through action bus 293, part of parameters bus 29, used to load the action parameters at power-up, from controller 21 (see Fig. 3).

Alarm/warning /action bus 28 is used to issue the alarm/warning if a suspect string or instruction was detected.

It is combined with similar bus from other comparator units (not shown) operating concurrently with that detailed in Fig. 4, and with their bus 28 connected in parallel to this one to form report alarm bus 241, to report to controller 21 that an alarm/warning or suspect activity took place.

Operation of the unit: This is a pipeline structure, with each instruction or sample of a long string being compared concurrently with all the other samples, using the comparator stages 331 ... 333. Thus, for each new instruction entered into register 22 from bus 17, that instruction together with the previous instructions, are compared with an instruction string pertaining to a virus or sensitive instruction stored in register 31, using mask in register 32.

The operation of the unit is synchronous, in that all the stages of register 22 are shifted at the same time, and the operation of the comparators is synchronized with that shift.

Thus, it was detailed how a string of bytes/ instructions are compared with a reference, by sliding the instructions intercepted from the PC busses past the fixed reference in register 31, using register 22.

What is compared, is shifted in SR 22 – only data or only instructions or also (optional) address or also (optional) control bus. Address information in register 22 allows to relate to specific address range, like interrupt vectors in RAM of PC, or to first instructions (from fixed address) after RESET.

Control information in register 22 allows to relate, or wait for specific instruction types, like only READ, or READ and I/O, or READ and MEMORY, or WRITE and I/O, or DMA, or other combinations.

Control of CPU in applications processor, examples for several CPUs:

1. For the Intel Model i486, or 80486, used in Personal Computers (PC) "compatibles":

RESET – the reset input forces the CPU to abandon all tasks, and restart from the beginning, as in power-up. Can be used where real danger is eminent, and no risks can be taken to preserve status or tasks in progress

HOLD – the hold input causes the CPU to float most of its output and input/output pins, thus allowing access to the supervisor to the address, control and data busses. That is, full control over the PC and its resources. After deactivating the HOLD signal, the CPU will continue its tasks as usual.

To float the outputs means that the outputs go into a tri-state or high impedance state, allowing other devices to control the logic state for the bits on the bus, either 1 or 0.

BOFF – the backoff input forces the CPU to float its bus in the next clock, having an effect similar to HOLD.

2. For the Intel Model Pentium, used in Personal Computers (PC) "compatibles":

RESET – the reset input forces the CPU to abandon all tasks, and restart from the beginning, as in power–up. Like in i486.

HOLD – the Hold Request input causes the CPU to float most of its output and input/output pins.

BOFF – the Back Off input forces the CPU to float its bus in the next clock, having an effect similar to HOLD.

BRDY – indicates that the external device is ready to transfer data.

3. For the Motorola Model 68020 CPU:

RESET – the reset input forces the CPU to abandon all tasks, and restart from the beginning, as in power–up.

HALT – this input stops all processor bus activity, and the bus is floated, to tri–state.

Mask bytes example, for Intel 8086/80486 family:

Structure of instruction "Jump within segment or group" :

Three bytes: Op Code, Disp Low , Disp High ;

Op Code= E9 hexa

Suppose that we aim to detect the "Jump" instruction, where the actual address is irrelevant. Maybe in different computers will load to different addresses. Then the first byte is enabled, and the next two bytes are completely disabled or masked out.

The result: any program including the "Jump" instruction of this type, with whatever parameters, will be detected reliably.

Mask bits example, for Intel 8086/80486 family:

Structure of instruction "Jump on condition" :

Two bytes: Op Code, Displacement ;

Op Code= 70 hexa Jump if overflow

Op Code= 71 hexa Jump if no overflow

Op Code= 72 hexa Jump if below

Op Code= 73 hexa Jump if above

Suppose we desire to detect all of these instructions with one comparator unit. The difference between the op codes is in the lowest two bits, (D1 D0) .

By using a mask 1111 1100 binary and an AND operation with the opcode, the lowest bits are masked out, to be always zero.

The masked opcode is the same for the four instructions, which can be detected with one comparator. Compare with 70 hexa.

If the displacement is irrelevant, then all its bits can be masked out as well (for example by performing AND with 0000 0000 binary).

Thus, means are detailed for the supervisor to detect viruses without slowing down the applications processor.

Fig. 5 details an embodiment of the comparator structure and operation. It details, for example, one embodiment of comparator 331, mentioned with reference to Fig. 4 above. Comparator 331 operates on busses C0 226, R0 316 and M0 326, each k bits wide latest output C0 226, the corresponding reference R0 316 and mask from bus M0 326.

Fig. 5 details the operations on one bit, say bit j. This is representative of the operation of the other bits as well.
One bit of the instruction C0(j) 412 is compared with one bit of the reference R0(j) 413, in XOR gate 41. It compares latest output C0 226 with corresponding reference R0 316, using AND mask from bus M0 326.

The comparison is bitwise, that is each of k bits of bus C0 226 is compared with a corresponding bit of bus R0 316, with mask bit from M0 326 being applied to allow or ignore the result, then all the partial bit results are collected to result in result 361.

The operations on one bit, say bit j, are detailed. This is representative of the operation of the other bits as well.
Thus, one bit of the instruction C0(j) 412 is compared with one bit of the reference R0(j) 413 in XOR gate 41. The output 414 of gate 41 is zero if the bits are equal (either both are 0 or both are 1 logic).
If the mask bit M0(j) 423 is HIGH or ENABLE, then AND gate 42 is transparent, so that its output 424 is identical to signal 414. Similarly, the other comparators (not shown) operate on the other bits of the busses C0, R0, M0 to issue the other input lines 425 to NOR gate 43.

If, and only if, all these input lines 424 and 425 are at ZERO state, then the output 361 of NOR gate 43 goes HIGH, indicating an ALARM or identity for that stage in comparison.

The above description relates to a bit for which the corresponding mask bit 423 is ENABLE, or M0(j)= HIGH. If the mask state is IGNORE or MASK, that is ZERO, then the output 424 of AND gate 42 is always ZERO, regardless of the result of the comparison in gate 41. Thus, the bit j is ignored and assumed equal to the reference, to allow the issuing of an ALARM if the other bits (those which are not ignored) are equal to the reference.

The result of comparison 361 at the output of NOR gate 43 is combined (not shown) with corresponding results from the other stages of comparison, to arrive at a decision regarding a specific virus pattern which is sought after.

The logic circuit of Fig. 5 and the corresponding explanation are just one embodiment of the bitwise comparison process.
Other embodiments will become apparent to persons in the art, like a PAL or FPLA or microcontrolled implementation to perform logical operation or a multiplexed comparison to perform sequential operation on the bits and/or the bytes or word of the relevant busses.

In another embodiment of the abovedetailed structure and method, a digital signal processor like the ANALOG DEVICES INC. DSP21000 family may be used to perform the bitwise logic and other functions as detailed above.

Fig. 6 details another structure of monitoring means, using digital signatures in lieu of direct comparison of all samples of an instruction string.

Data multiplexer and latches means 222 reads instructions from the input bus and generates instruction/data bus 223, which is transferred to comparator shift register 22, and thence to log shift register/FIFO 23. These registers shift all the time, to move the instructions past the comparation unit. The log shift register/FIFO output bus 231 is connected to the controller (not shown).

Thus, for example, register 22 outputs data bus latest output C0 226, and previous samples including data bus i-th output Ci 227 which represents one of the n stages in SR 22, and data bus earliest output Cn 228.

There are also a multitude of comparator units operating concurrently, each receiving the outputs from register 22 in parallel, and each including:

1. signature computing means 51, which receives a data bus 512 from SR 22 and a mask bus 513 from a register (not shown). The output is issued on the digital signature bus 515.

2. comparator 52, compares signature on bus 514 with reference of virus or other event, stored in signature reference register 53 . This is a fixed value, after initial loading. It is loaded through load signature reference bus 291, part of parameters bus 29.

The mask register (not shown) is serial or parallel, as known in the art. It is fixed after initial loading. An alarm strobe 335 is issued (active) if the signatures correspond, indicating the complete detection of a virus string. The alarm strobe 335 connects to an action register (not shown) similar to that in Fig. 4, or similar.

The comparator unit includes a plurality of such comparators, each with a signature computing means 51, comparator 52 and reference 53, and each issuing an alarm strobe 335.

Each may issue a different type of Alarm or Warning or Notice, as programmed by the controller (not shown) in the supervisor processor. All the comparators operate concurrently on the data/instructions supplied to all of them, in parallel, from SR 22.

Example of use of digital signatures:

Time to error – for 200 MHz clock, that is, one comparison and decision is performed each 5 nanoseconds interval.

Time to error: until the accumulated probability of error Pr is Pr=0.5. That is the probability for at least one error (the probability that not a single error occurred is too = 0.5), see Table 1.

Table 1

Number of bytes	Max. value 2^n	Probability of one error	Time to error, seconds	Time to error, years
2	65536	1.52E-05	1.63E-04	5.19E-12
3	1.67E+07	5.96E-08	4.19E-02	1.33E-09
4	4.29E+09	2.32E-10	11.	3.40E-07
5	1.09E+12	9.09E-13	2748.	8.71E-05
6	2.81E+14	3.55E-15	703687.	2.23E-02
7	7.20E+16	1.38E-17	1.80E+08	5.7
8	1.84E+19	5.42E-20	4.61E+10	1462.

Thus, using a signature of eight bytes, the 50% cumulative probability of error is achieved after continuous operation for 1,462 years, at a clock of 200 Megahertz. Thus signatures may be used to achieve a very reliable verification, while concurrently achieving a significant saving in cost and circuit complexity.

According to another aspect of the present invention, the supervisor includes a multisensor unit 27 which monitors everything going on, in and around the application processor, including (not shown) a plurality of sensors for sensing the various effects indicative of the presence of a virus.

The multisensor unit– monitors everything going on, including processor commands, I/O port activity, RS–232, modem/telephone output, ethernet, Internet, LAN, optical (visible/IR/UV), ultrasonic/sonic/subsonic, RF and microwaves, electrical and magnetic fields, disturbance on power line of 50 Hz or 60Hz. Its function: look everywhere, every medium is suspect; anything moving, bring in for inspection.

Part of the sensors are fixed, to continuously sense in and around the PC. Part of the sensors are portable, detachable, like a pen for example, so as to allow the user to bring it to suspect areas or busses.

Various implementations and other embodiments of the present invention will become apparent to persons skilled in the art upon reading the abovedetailed disclosure.

It will be recognized that the foregoing is but one example of an apparatus and method within the scope of the present invention and that various modifications will occur to those skilled in the art upon reading the disclosure set forth hereinbefore.

Claims

What is claimed is:

1. An antivirus protection system comprising a second processor attached to a first processor for monitoring its performance and intervening if predefined behaviour thereof is detected, wherein said second processor includes means for continuously supervising the operation of the first processor to detect virus-related activities therein by comparing actual performed instructions in the first processor with instruction sequences corresponding to known viruses or to predefined suspect behaviour therein, and to stop said activity in the first processor after its detection.

2. The antivirus system according to claim 1, further including means for protection from attack by virus in said first processor and including a separate computer and memory, which are independent from said processor.

3. The antivirus system according to claim 1, wherein said supervising means includes a combination of fast monitoring means in hardware, with smart verification means in a programmed device, with these means operating together to protect from known viruses as well as from unknown viruses according to their suspect behavior.

4. The antivirus system according to claim 1, further including a plurality of sensors to detect suspect activity in various media like radio frequency or wireless RF, serial or parallel communication channels, ultrasound, sonic or subsonic waves, power lines and/or optical media including infrared, visible light and/or ultraviolet.

5. An antivirus multiprocessor system comprising a first processor for performing application programs and a second processor for continuously supervising the operation of the first, wherein said second processor is connected to the bus of said first processor and includes means to monitor activity of the first processor to detect virus-related activities therein by comparing actual performed instructions in the first processor with instruction sequences corresponding to known viruses or to predefined suspect behaviour therein, and to stop said activity in the first processor after its detection.

6. The antivirus system according to claim 5, further including means for protection from attack by virus in the first processor and including a separate computer and memory, which are independent from said processor.

7. The antivirus system according to claim 5, wherein said supervising means includes a combination of fast monitoring means in hardware, with smart verification means in a programmed device, with these means operating together to protect from known viruses as well as from unknown viruses according to their suspect behavior.

8. The antivirus system according to claim 5, further including a plurality of sensors to detect suspect activity in various media like radio frequency or wireless Rf, ultrasound, sonic or subsonic waves, power lines and/or optical media including infrared, visible light and/or ultraviolet.

9. In a multiprocessor system, an antivirus protection method including the steps of:

A. connecting a second processor to a first applications processor;

B. using the second processor to continuously monitor the activities performed on the first processor and to compare to predefined forbidden activities; and

C. when the second processor detects a forbidden activity taking place in the first processor, then the second processor takes predefined actions to prevent damage to the second or applications processor.

10. The antivirus protection method according to claim 9, further including in step (B) the monitoring by the second processsor of a multisensor unit connected thereto.

11. The antivirus protection method according to claim 10, wherein the multisensor unit includes a plurality of sensors to detect suspect activity in various media like radio frequency or wireless Rf, ultrasound, sonic or subsonic waves, power lines and/or optical media including infrared, visible light and/or ultraviolet.

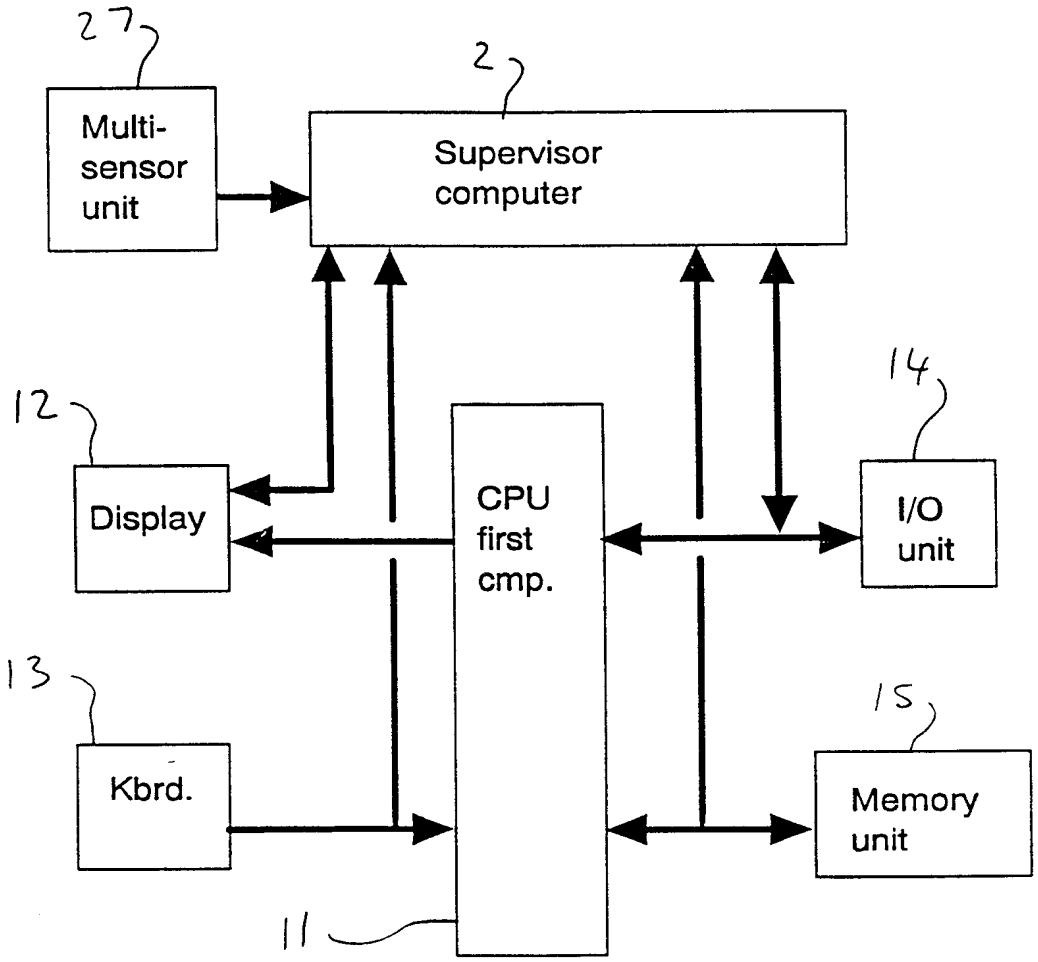

27
Multi-
sensor
unit
2
Supervisor
computer
12
Display
CPU
first
cmp.
14
I/O
unit
13
Kbrd.
15
Memory
unit
11

Fig. 1

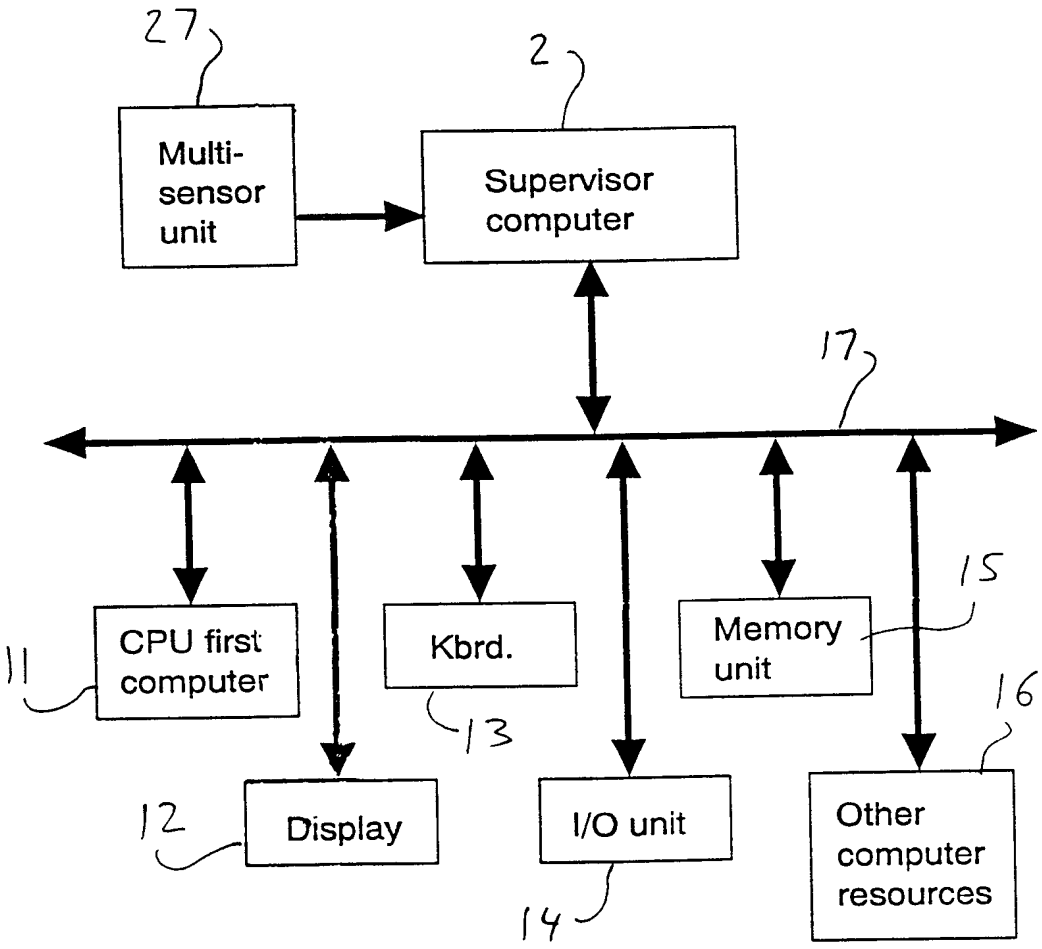

27
2
Multi-sensor unit
Supervisor computer
17
15
CPU first computer
11
Kbrd.
13
Memory unit
16
12
Display
I/O unit
14
Other computer resources

Fig. 2

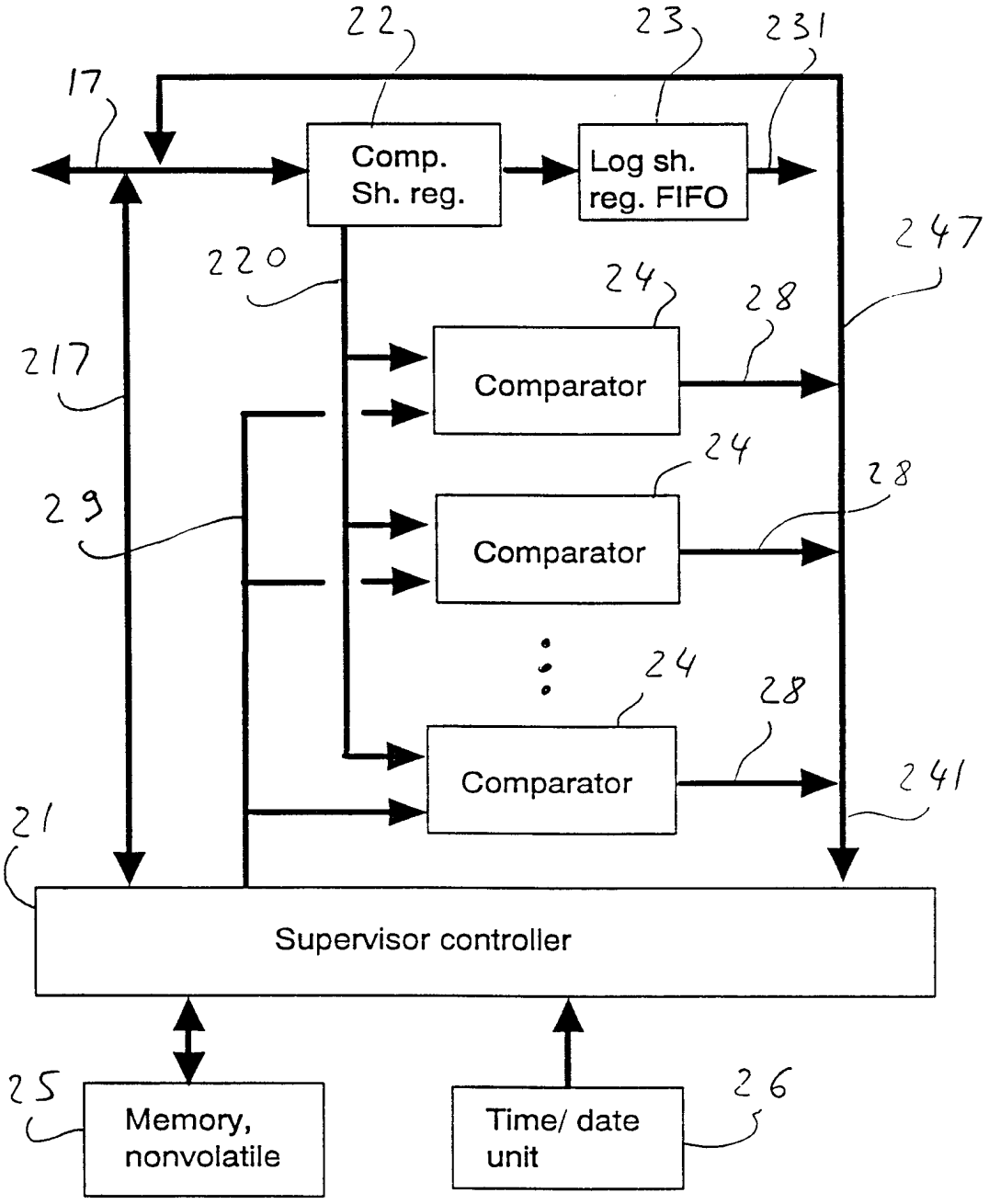
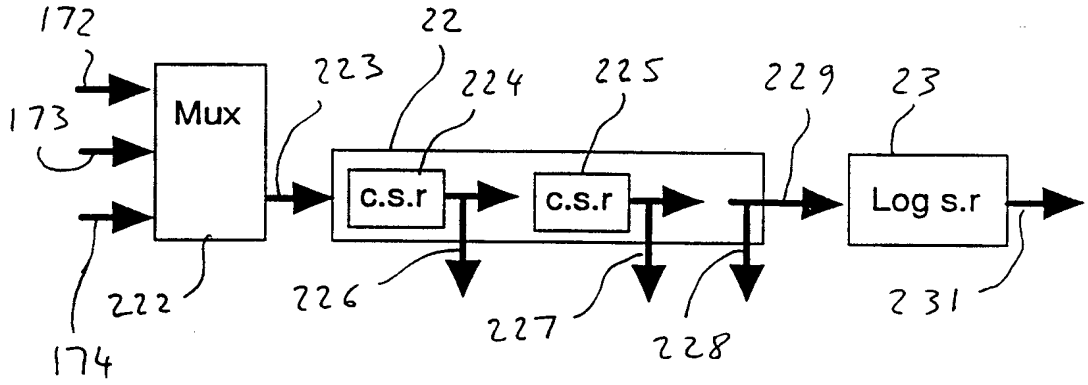

22
23
231
17
Comp.
Sh. reg.
Log sh.
reg. FIFO
247
220
24
28
217
Comparator
24
28
29
Comparator
24
28
Comparator
241
21
Supervisor controller
25
Memory,
nonvolatile
Time/ date
unit
26

Fig. 3


172
173
174
Mux
222
223
22
224
225
c.s.r
c.s.r
226
227
228
229
23
Log s.r
231

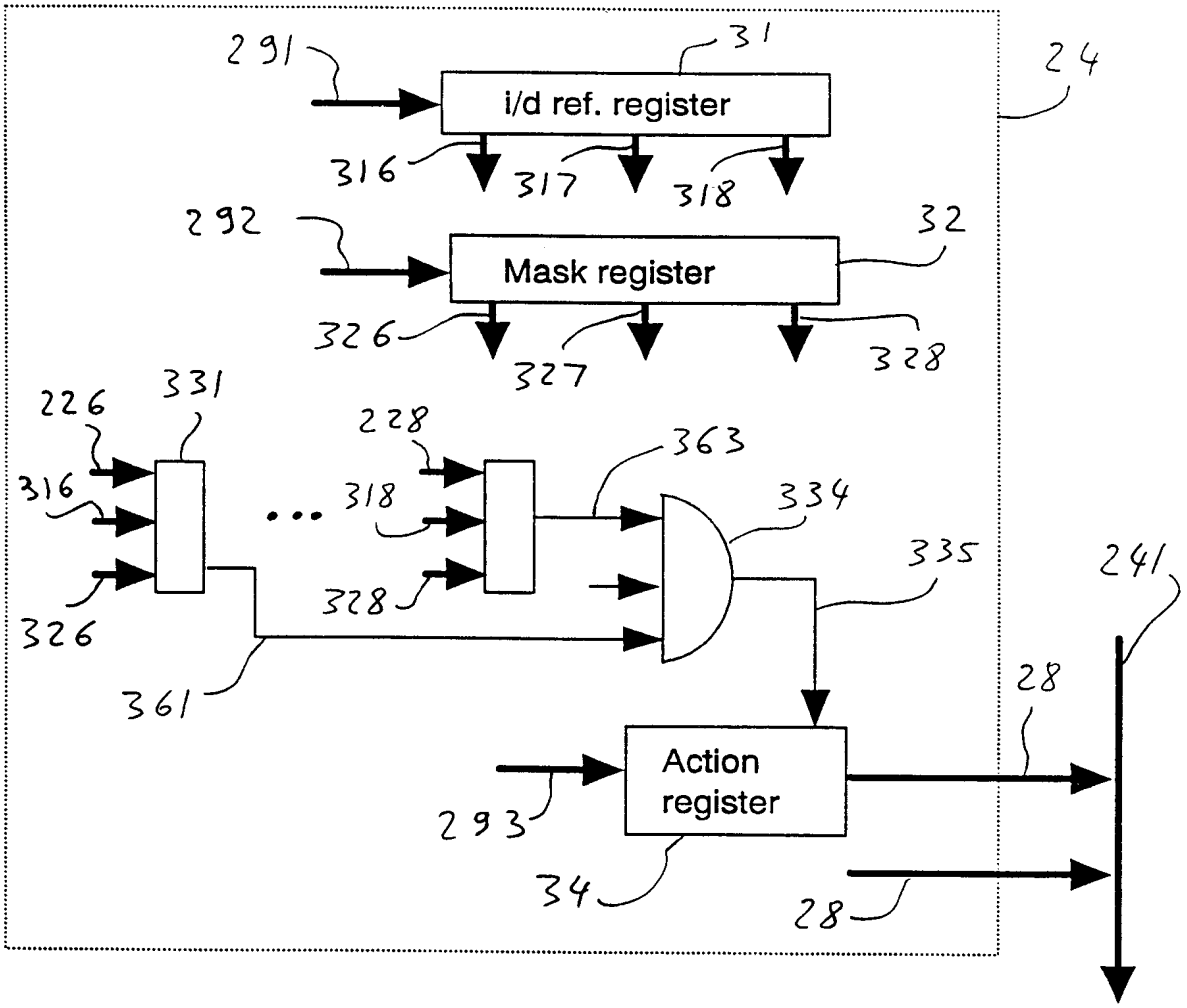

291
31
24
i/d ref. register
316
317
318
292
32
Mask register
326
327
328
226
331
316
326
228
318
328
363
334
335
241
361
28
Action register
293
34
28

Fig. 4

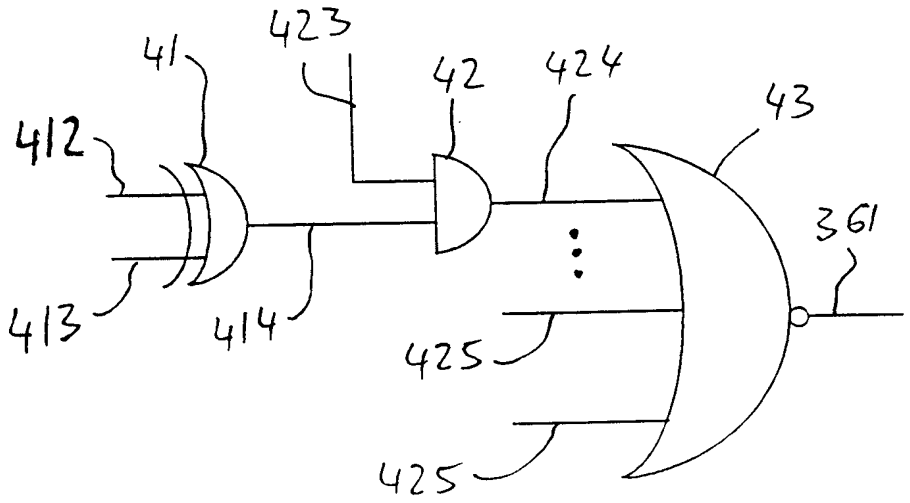


Fig. 5

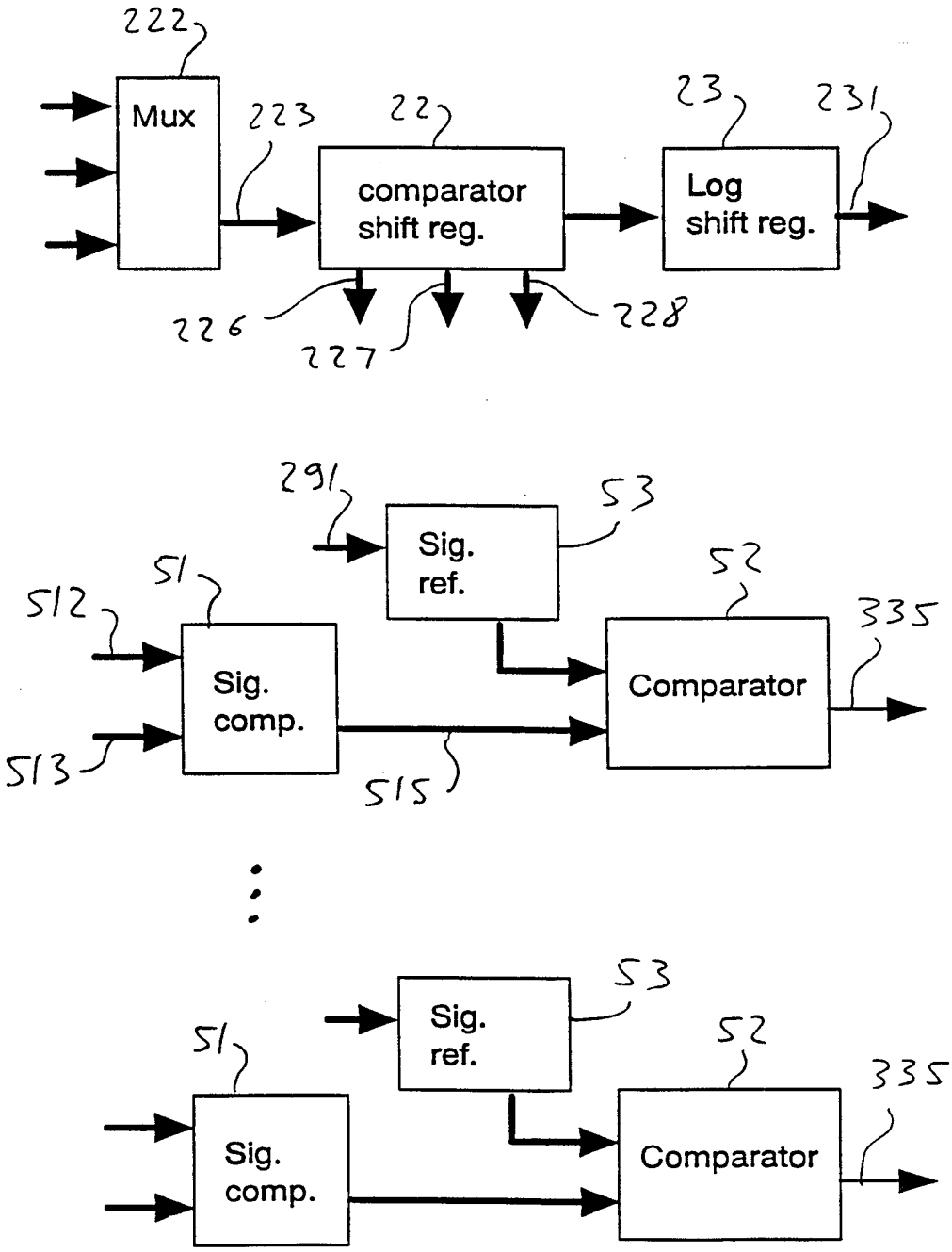

222
Mux
223
22
comparator
shift reg.
23
231
Log
shift reg.
226
227
228
291
53
Sig.
ref.
52
335
51
512
Sig.
comp.
513
515
Comparator
53
Sig.
ref.
51
52
335
Sig.
comp.
Comparator

Fig. 6